Олимпиада школьников

ТИИМ - Технологии. Интеллект. Информатика. Математика

(Задания отборочного тура по информатике. 2020-2021 учебный год)

14 мая 2021 г.

Олимпиада по информатике

Задания отборочного тура олимпиады с решениями

Олимпиада ТИИМ по информатике проводилась дистанционно, с применением системы автоматического тестирования решений учащихся на наборах тестовых данных. Каждая из 6 задач максимально оценивалась в 10 баллов, в соответствии с количеством успешно пройденных тестов. Задания были рассчитаны на учащихся 8-11 классов, все учащиеся решали один вариант. На решение отводилось 5 часов, как для отборочного, так и для заключительного тура.

Задача 1. Буратино идет к успеху

Буратино продал букварь и решил выгодно инвестировать вырученные средства Он что-то слышал про поле чудес, но не знал, как туда попасть.
После долгих поисков в интернете он нашел два объявления:
Поле чудесных кубов
Закопайте свои денежки на нашем поле и каждый день каждая цифра суммы на счете будет умножаться сама на себя дважды, после чего полученные числа будут сложены.
*Например, если вы положите на счет всего 99 монет , уже на второй день сумма будет равна 9*9*9+9*9*9=1458 монетам!*

Приходите к нам! С уважением, Алиса

Поле чудесных разностей
Передайте свои монеты к нам в доверительное управление и каждый день мы будем формировать из цифр вашего счета два числа - максимальное и минимальное. И на следующий день Вы получите разность этих двух чисел.
Например, у Вас было 1279 монет - на следующий день Вы получаете 9721-1279 = 8442 монеты

Откройте вклад у нас! С уважением, Базилио

Подскажите Буратино, куда лучше инвестировать

Входные данные
стоимость букваря, количество дней для инвестиций
выходные данные:
Один из трех вариантов:
“Поле чудесных кубов выгоднее на (сумма)”
“Поле чудесных разностей выгоднее на (сумма)”
“Буратино, не делай этого”, если оба варианта не принесут прибыли
Примечание:
Буратино планирует жить вечно, поэтому срок инвестиций в днях может быть большим, до 10^7.
Ограничение времени выполнения программы: 1 секунда
Набор тестовых данных:

Входные данные	Результат работы программы
1 1000000	Буратино, не делай этого
99 1	Поле чудесных кубов выгоднее на 1458
1000 2	Буратино, не делай этого
1001 2	Поле чудесных разностей выгоднее на 9613
2 10	Поле чудесных кубов выгоднее на 371
7896 500	Буратино, не делай этого
2322 4	Буратино, не делай этого
2792 100	Поле чудесных разностей выгоднее на 5803
1921 999999	Поле чудесных разностей выгоднее на 4715
1921 1000000	Поле чудесных разностей выгоднее на 5255

Решение задачи на языке Python

```
#Функция для вычисления суммы кубов цифр
def cube(n):
    num_list = map(int, list(str(n)))
    n = list(map(lambda a:a*a*a, num_list))
    return sum(n)

#Функция для вычисления разности чисел — наибольшего,
#составленного из цифр этого числа, и наименьшего,
#составленного из цифр этого числа
def diff(n):
    num_list = list(str(n))
    n = int(''.join(sorted(num_list,reverse = True)))
    -   int(''.join(sorted(num_list)))
    return n

#Функция, вычисляющая результат поля чудесных кубов
#за days дней
```

```
def cubes(n,days):
    for i in range(0,days):
        n1 = cube(n)
  #Если предыдущее число совпадает с вычисленным
  #на этом этапе – дальше можно не вычислять
        if n1==n:
            return n
        else:
            n=n1
    return n

#Функция, вычисляющая результат поля чудесных
#разностей за days дней
def diffs(n,days):
    for i in range(0,days):
        n1 = diff(n)
  #Если предыдущее число совпадает с вычисленным
  #на этом этапе – дальше можно не вычислять
        if n1==n:
            return n
        else:
            n=n1
    return n

#Функция, проверяющая выгоду Буратино
def solve(money,days):

    money_cube = cubes(money,days)
    money_diff = diffs(money,days)

    if(money_cube>money_diff and money_cube>money):
        return 'Поле чудесных кубов выгоднее на '
        +str(money_cube-money_diff)

    if(money_diff>money_cube and money_diff>money):
        return 'Поле чудесных разностей выгоднее на '
        +str(money_diff-money_cube)

    return 'Буратино, не делай этого'

money = int(input())
days = int(input())

print(solve(money,days))
```

Задача 2. Рождение Смауга

Дракон Смауг, охраняющий золото гномов, решил установить в Одинокой Горе систему "умный дом"для защиты от несанкционированного проникновения.

Смауг решил не делать сложный код доступа, потому что он никогда не покидает пещеру.

Легенда гласит, что пароль доступа в сокровищницу меняется раз в год, на день рождения дракона, ровно в 0 часов 0 минут и генерируется следующим образом: день, на который выпал день рождения Смауга в текущем году(Mon,Tue, Wed, Thu, Fri, Sat,Sun), за ним идут цифры - количество раз, которое он отмечал свой день рождения в этот день.

Кроме того, Смауг очень гордится своим возрастом и опытностью и любит большие числа. Поэтому у главного входа в Одинокую Гору он повесил огромное световое табло, которое может показывать до 12 цифр, чтобы показывать на нем свой текущий возраст в секундах

Бильбо Беггинсу очень нужно попасть в пещеру. Помогите Бильбо подобрать код

Входные данные: день рождения Смауга в формате дд.мм.гггг Число на табло

Выходные данные:

код доступа

Ограничение времени выполнения программы: 1 секунда
Набор тестовых данных:

Входные данные	Результат работы программы
02.01.1921 3162240000	Sat15
15.01.2021 30000000	Fri1
09.05.2000 1000	Tue1
29.02.2000 9999999999	Tue11
29.02.2000 30000000	Tue1
01.01.0001 31622400000	Sat141
01.01.2001 316224000000	Fri1453
09.05.1050 2000000	Thu1
01.01.0001 9999999999	Mon45
08.03.9999 1000000000	Fri5

Решение задачи на языке Python

```
import datetime


def count_birthdays(last_birth_wd, d,m,start_year,end_year):
    c = 0
    for y in range(start_year, end_year):
        try:
            if datetime.datetime(y,m,d).weekday() == last_birth_wd:
                c += 1
        except:
            c += 0
```

```
    return c

def main():
    birth_date = input()
    seconds = int(input())

    (d,m,birth_year) = map(int,birth_date.split('.'))
    """Грегорианский календарь образует цикл в 400 лет, а значное12— число
    секунд может быть более 30 тыс лет. """

    """Год рождения берем равным остатку от деления года рождения на 400
    лет, чтобы не было шансов при вычислении перевалить за 9999 год"""
    birth_year %= 400
    fourhundredyears = int((datetime.datetime(2400,1,1)-datetime.
    datetime(2000,1,1)).total_seconds())
    repeats = seconds // fourhundredyears


    days = ['Mon','Tue', 'Wed', 'Thu', 'Fri', 'Sat','Sun']
    birth_date = datetime.datetime(birth_year,m,d)

    cur_date = birth_date + datetime.timedelta(0,seconds%
    fourhundredyears)

    this_year = cur_date.year


    try:
        this_year_birth_date = datetime.datetime(this_year,m,d)
    except:
        this_year_birth_date = datetime.datetime(this_year-1,m,d)

    if this_year_birth_date <= cur_date:
        last_birth_date = this_year_birth_date
    else:
        last_birth_date = datetime.datetime(this_year-1,m,d)

    last_birth_wd = last_birth_date.weekday()


    count = count_birthdays(last_birth_wd,d,m,birth_year,this_year)
    count_repeated = count_birthdays(last_birth_wd,d,m,birth_year,
    birth_year+400)


    print(days[last_birth_wd]+str(1+count+count_repeated*repeats))

main()
```

Задача 3. Препятствие для королевы

Гермиона и Гарри решили сбежать к Хагриду, чтобы поиграть с Пушком, и придумали для Рона отвлекающую задачу.

Гермиона выбирает размер шахматного поля n, Гарри располагает на нем вымышленную фигуру “Философский камень”. Задача Рона - найти все возможные расположения n ферзей на поле, таких, чтобы ни один ферзь не атаковал другого.
Фигура “Философский камень” не ходит, не атакует, но через нее ферзь не может атаковать другого ферзя, и на его месте не может стоять другая фигура.
На какое время Рон будет занят этой задачей, если на одну расстановку ферзей на поле, удовлетворяющую условиям, у него уходит 1 минута?
Входные данные:
Размер доски, 3<=n<=10
координаты философского камня(c,r) 0<=r<n, 0<=c<n
Выходные данные:
количество минут, на которые Рон будет занят
Ограничение времени работы программы: 1 с

Набор тестовых данных:

Входные данные	Результат работы программы
3 1 0	1
3 0 0	1
4 0 1	2
5 2 1	10
6 3 3	30
6 2 3	30
5 3 2	10
7 0 0	36
8 0 0	88
8 6 1	206

Решение задачи на языке C++. Автор решения - победитель ТИИМ по информатике, Иван Захаров, 10 класс, г. Москва

```
#include <bits/stdc++.h>

#define all(a) (a).begin(), (a).end()
#define rall(a) (a).rbegin(), (a).rend()
#define sz(a) (int)(a).size()

using namespace std;

const int MAXN = 10;

int x, y, n;
int used[MAXN][MAXN];
```

```
long long cnt;

inline void go(int ind, int cnt, int flag, int start) {
    if (ind > y && n - ind < n - cnt) {
        return;
    }

    if (n - ind < n - cnt - 1) {
        return;
    }


    if (ind == n || cnt == n) {
        if (cnt == n) {
            /* cout << "used = \n";
            for (int i = 0; i < n; i++) {
                for (int j = 0; j < n; j++) {
                    cout << used[i][j] << " ";
                }
                cout << "\n";
            }
            cout << "\n"; */

            ::cnt++;
        }
        return;
    }

    if (!flag) {
        go(ind + 1, cnt, 0, 0);
    }

    for (int k = start; k < n; k++) {
        if (used[k][ind] || (k == x && ind == y)) {
            continue;
        }

        used[k][ind] = 10;

        for (int i = ind + 1; i < n && !(k == x && i == y); i++) {
            used[k][i]++;
        }
        for (int i = ind - 1; i >= 0 && !(k == x && i == y); i--) {
            used[k][i]++;
        }
        for (int i = k + 1; i < n && !(i == x && ind == y); i++) {
            used[i][ind]++;
        }
        for (int i = k - 1; i >= 0 && !(i == x && ind == y); i--) {
            used[i][ind]++;
        }

        for (int i = 1; k + i < n && ind + i < n &&
```

```
  !(k + i == x && ind + i == y); i++) {
    used[k + i][ind + i]++;
}
for (int i = 1; k - i >= 0 && ind - i >= 0 &&
  !(k - i == x && ind - i == y); i++) {
    used[k - i][ind - i]++;
}

for (int i = 1; k + i < n && ind - i >= 0 &&
  !(k + i == x && ind - i == y); i++) {
    used[k + i][ind - i]++;
}
for (int i = 1; k - i >= 0 && ind + i < n &&
  !(k - i == x && ind + i == y); i++) {
    used[k - i][ind + i]++;
}

if (ind == y && flag == 0 && k < x && cnt + 1 < n) {
    go(ind, cnt + 1, 1, k + 1);
}

go(ind + 1, cnt + 1, 0, 0);

used[k][ind] = 0;

for (int i = ind + 1; i < n && !(k == x && i == y); i++) {
    used[k][i]--;
}
for (int i = ind - 1; i >= 0 && !(k == x && i == y); i--) {
    used[k][i]--;
}
for (int i = k + 1; i < n && !(i == x && ind == y); i++) {
    used[i][ind]--;
}
for (int i = k - 1; i >= 0 && !(i == x && ind == y); i--) {
    used[i][ind]--;
}

for (int i = 1; k + i < n && ind + i < n &&
  !(k + i == x && ind + i == y); i++) {
    used[k + i][ind + i]--;
}
for (int i = 1; k - i >= 0 && ind - i >= 0 &&
  !(k - i == x && ind - i == y); i++) {
    used[k - i][ind - i]--;
}

for (int i = 1; k + i < n && ind - i >= 0 &&
  !(k + i == x && ind - i == y); i++) {
    used[k + i][ind - i]--;
}
for (int i = 1; k - i >= 0 && ind + i < n &&
  !(k - i == x && ind + i == y); i++) {
```

```
                used[k - i][ind + i]--;
            }
        }
    }

signed main() {
    ios_base::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);

    memset(used, 0, sizeof(used));

    cin >> n >> x >> y;
    cnt = 0;
    go(0, 0, 0, 0);

    cout << cnt << "\n";
    return 0;
}
```

Задача 4. Детские забавы

У Пети есть младший брат, Лёва. Лёве три года, он очень смышленый, умеет считать до четырех, говорит, что мечтает стать программистом, как Петя, и его любимая игра - собирать квадрат 4*4 из своего бесконечного набора деталей. Он может заниматься этим часами.

Набор состоит из деталей размером 1*1, 1*2, 1*3 и 1*4, каждой детали неограниченное количество.

Детали могут размещаться горизонтально, но не вертикально, т.е. вариант

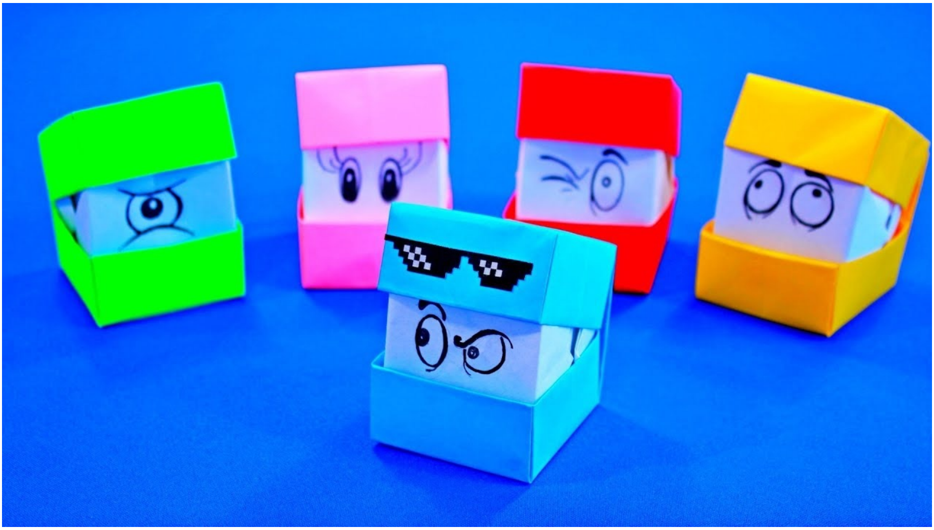
AAAB AAAB AAAB AAAB, где BBBB - деталь 1*4, а AAA - детали 1*3, не допустим

Лёва придумал новую игру. Он говорит Пете, какие детали из набора взял, а Петя угадывает, сколько квадратов Лёва из них собрал.

Составьте программу, которая будет на основе списка, сообщенного Лёвой, выдавать количество собранных квадратов

Входные данные:

Список размеров деталей, разделенный пробелами, например 1 4 2 3 4 2



Выходные данные:

количество квадратов, которые можно собрать

Ограничение времени работы программы: 1 с

Набор тестовых данных:

Входные данные	Результат работы программы
4 4 4 4 4 4 4 4	2
3 1 2 2 4 3 1 4 2 3 1 2 4 3 1 2 2 4 3 1 4 2 3 1 2 4	4
1 1 1 2 2 2 2 3 3 3 4 4 4 1 1 1 2 2 2 2 3 3 3 4 4 4	4
3 3 3 3 3 1 1 1 2 2	1
3 3 3 3 3 3	0
4 4	70
4 4 4 4 2 2 2 2 2 2 2 2 3 3 3 3 1 1 1 1 1	3
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2	4
1 1 2 3 4	0

Решение задачи на языке Python:

```
def main():
```

```
    details = input().split()
    sets=[0,0,0,0,0]
    for i in range(1,5):
        for detail in details:
            if int(detail) == i:
                sets[i] +=1

    """Количество строк, заполненных деталями 1*4"""
    rows = sets[4]

    """Количество строк, заполненных деталями 1*3 и 1*1"""
    if sets[3]>0:
        if sets[1]>=sets[3] :
            rows+=sets[3]
            sets[1] -= sets[3]
        else:
            rows+=sets[1]
            sets[1] = 0

    """Количество строк, заполненных деталями 2*1 и 2*1"""
    if sets[2]>0 :
        rows+=sets[2]//2
        sets[2] = sets[2] % 2
        """Количество строк, заполненных деталями 2*1 1*1 и 1*1"""
        if sets[2]>0 :
            if sets[1]>1:
                rows+=1
                sets[1]-=2

    """Количество строк, заполненных деталями 1*1"""
    if sets[1]>0 :
        rows+=sets[1]//4

    print (rows // 4)

main()
```

Задача 5. Помогите Даше найти выход

Даша-следопыт пришла в себя в очень странном черно-белом месте. Вокруг нее были только стены, и какой-то слабо освещенный коридор.

В кармане она нашла кнопочный телефон с одним странным и очень длинным сообщением. Сообщение содержало подозрительно большое количество пробелов в неожиданных местах и точки на одинаковом расстоянии друг от друга:

```
DearDasha!Iknowyo.@       u       a.r escar e d,but e.v   ery t     h i.n g w i llbef
i n.e   j u       s t.fol l owmyins t r.u   c t    i  o n.s bes t re g ar *.dsyourmonkey#
```

Даша поняла: это зашифрованная карта лабиринта!

```
DearDasha!Iknowyo
@       u       a
r escar e d,but e
v   ery t     h i
n g w i llbef i n
e   j u       s t
fol l owmyins t r
u   c t    i  o n
s bes t re g ar *
dsyourmonkey#####
```

```
##################
@       #        #
# ##### # ##### #
#   # # #     # #
# # # # ##### # #
#   # #       # #
### # ####### # #
#   # #    #  # #
# ### # ## # ## *
##################
```

Поможем Даше-следопыту найти путь из лабиринта!
Входные данные:

Текстовая строка
Пробел - проход в лабиринте
Точка - перенос строки
@ - начальное положение Даши
* - место, куда надо прийти
Любой другой символ - стены

Выходные данные:
путь Даши по шагам в виде строки из символов > | <^(вправо вниз влево вверх)

Ограничение времени работы программы: 1 с

Пример тестовых данных:

```
##################.@       #       #.# ##### # ##### #.#   # # #     # #.# # # # #####
# #.#   # #       # #.### # ####### # #.#   # #    #  # #.# ### # ## # ## *.##############
```

Результат работы программы

```
>>>>>>>||||>>>>>>>^^<<<<^^>>>>>>|||||||>
```

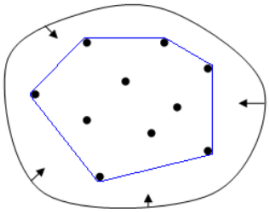
Решение задачи на языке Python:
Автор решения - победитель заключительного тура олимпиады по информатике, Артем Шмараев, 10 класс, г. Белгород

```
def found(pathArr, finPoint):
    weight = 1
    for i in range(len(pathArr) * len(pathArr[0])):
        weight += 1
        for y in range(len(pathArr)):
            for x in range(len(pathArr[y])):
                if pathArr[y][x] == (weight - 1):
                    if y > 0 and pathArr[y - 1][x] == 0:
                        pathArr[y - 1][x] = weight
                    if y < (len(pathArr) - 1) and pathArr[y + 1][x]
    == 0:
                        pathArr[y + 1][x] = weight
                    if x > 0 and pathArr[y][x - 1] == 0:
                        pathArr[y][x - 1] = weight
                    if x < (len(pathArr[y]) - 1) and pathArr[y][x +
    1] == 0:
                        pathArr[y][x + 1] = weight

                    if (abs(y - finPoint[0]) + abs(x - finPoint[1]))
     == 1:
                        pathArr[finPoint[0]][finPoint[1]] = weight
                        return True
    return False


def printPath(pathArr, finPoint):
    y = finPoint[0]
    x = finPoint[1]
    weight = pathArr[y][x]
```

```
    result = list(range(weight))
    while (weight):
        weight -= 1
        if y > 0 and pathArr[y - 1][x] == weight:
            y -= 1
            result[weight] = '|'
        elif y < (len(pathArr) - 1) and pathArr[y + 1][x] == weight:
            result[weight] = '^'
            y += 1
        elif x > 0 and pathArr[y][x - 1] == weight:
            result[weight] = '>'
            x -= 1
        elif x < (len(pathArr[y]) - 1) and pathArr[y][x + 1] ==
    weight:
            result[weight] = '<'
            x += 1

    return result[1:]


a = input().split(".")
for i in range(len(a)):
    a[i] = list(a[i])
    for j in range(len(a[i])):
        if a[i][j] == " ":
            a[i][j] = 0
        elif a[i][j] == "@":
            pozIn = (i, j)
            a[i][j] = 0
        elif a[i][j] == "*":
            pozOut = (i, j)
            a[i][j] = 0
        else:
            a[i][j] = 1

labirint = a
path = [[x if x == 0 else -1 for x in y] for y in labirint]
path[pozIn[0]][pozIn[1]] = 1;

if not found(path, pozOut):
    print("Путь не найден!")

result = printPath(path, pozOut)
print("".join(result))
```

Задача 6. Захват территории

После длительного пути кочевое племя наконец пришло к осознанию, что им нужно поменять образ жизни на более оседлый. Они нашли подходящий для постоянной жизни участок. Чтобы быстрее разбить лагерь, решено было использовать в качестве опор ограждения уже стоящие на участке деревья, а сами ограждения располагать по прямой линии между опорами.



Какова максимальная площадь, которая может получиться в результате, и сколько метров ограждений требуется поставить?
Расположение деревьев задано их координатами на плоскости

Входные данные:
Координаты деревьев

Выходные данные:
Площадь и периметр в метрах, разделенные пробелом. С точностью до 4 знаков после запятой

Ограничение времени работы программы: 1 с

Набор тестовых данных:

Входные данные	Результат работы программы
0 0 -1 -1 0 -1 -1 0	1.0 4.0
-0.5 -0.5 -0.5 1 1 -0.5 0 0 -0.3 -0.3 0 0.1	1.125 5.1213
0 0 1 1 -1 0	0.5 4.6503
-10 -10 2 2 1 3 7 8 -5 -3 -2 4 -3 2 3 -5 0 0 4 7 10 10 10 -10 -10 10	400.0 80.0
2 2 1 3 7 8 -5 -3 -2 4 -3 2 3 -5 0 0 4 7	83.5 39.3394

Решение задачи на языке Python:
Автор решения - призер отборочного тура олимпиады по информатике, Артём Севидов, 10 класс, г. Реутов

```
m=[]
while True:
    try:
        b,c=map(float,input().split())
        m.append((b,c))
    except:
        break
m=list(set(m))
n=len(m)
if n<3:
    print(0,0)
else:
    t=0
    for i in range(n):
        if m[t][0]>m[i][0]:
            t=i
    s=[]
    for i in range(n):
        if i!=t:
            s.append(i)
    def g(h,k,l):
        return (k[0]-h[0])*(l[1]-k[1])-(k[1]-h[1])*(l[0]-k[0])
    def q(a,b):
        global m
        h=[]
        i=0
        j=0
        while i<len(a) and j<len(b):
            if g(m[t],m[a[i]],m[b[j]])>0:
                h.append(a[i])
                i+=1
            else:
                h.append(b[j])
                j+=1
        h.extend(a[i:])
        h.extend(b[j:])
        return h
    def f(a):
        global m
        if len(a)<=1:
            return a
        k=len(a)//2
        return q(f(a[:k]),f(a[k:]))
    s=f(s)
    y=[t,s[0]]
    for i in range(1,len(s)):
        while g(m[y[-2]],m[y[-1]],m[s[i]])<0:
            del y[-1]
        y.append(s[i])
    s=0
    for i in range(len(y)-1):
        s+=m[y[i]][0]*m[y[i+1]][1]
    for i in range(1,len(y)):
```

```
    s-=m[y[i]][0]*m[y[i-1]][1]
s+=m[y[-1]][0]*m[y[0]][1]
s-=m[y[0]][0]*m[y[-1]][1]
p=0
for i in range(len(y)-1):
    p+=((m[y[i]][0]-m[y[i+1]][0])**2+(m[y[i]][1]-m[y[i+1]][1])
**2)**0.5
p+=((m[y[0]][0]-m[y[-1]][0])**2+(m[y[0]][1]-m[y[-1]][1])**2)
**0.5
print(round(abs(s/2),4),round(p,4))
```